Summarizing Data

- You can use an aggregate function (or summary function) to produce a statistical summary of data in a table.

Function	Definition
AVG, MEAN	mean or average of values
COUNT, FREQ, N	number of nonmissing values
CSS	corrected sum of squares
CV	coefficient of variation (percent)
MAX(MIN)	largest(smallest) value
NMISS	number of missing values
PRT	probability of a greater absolute value of Student's t
RANGE	range of values
STD	standard deviation
STDERR	standard error of the mean
SUM	sum of values
SUMWGT	sum of the WEIGHT variable values
T	Student's t value for testing the hypothesis that the population mean is zero
USS	uncorrected sum of squares
VAR	variance

- Overview of Summarizing Data
  - You can use an aggregate function (or summary function) to produce a statistical summary of data in a table.
  - You can specify one column or more than one columns.
  - If you specify one column as the argument to an aggregate function, then the values in that column are calculated.
  - When more than one argument is used within an SQL aggregate function, the function is no longer considered to be an SQL aggregate or summary function.
  - If there is a like-named Base SAS function, then PROC SQL executes the Base SAS function and the results that are returned are based on the values for the current row. If no like-named Base SAS function exists, then an error will occur. For example, if you use multiple arguments for the AVG function, an error will occur because there is no AVG function for Base SAS.

- More than one column for argument

```
proc sql outobs=12;
    title 'Mean Temperatures for World Cities';
    select City, Country, mean(AvgHigh, AvgLow) as MeanTemp
    from sql.worldtemps
    where calculated MeanTemp gt 75
    order by MeanTemp desc;
quit;
```

  - Note:You must use the CALCULATED keyword to reference the calculated column.

- One column argument

```
proc sql;
    title 'World Oil Reserves';
    select sum(Barrels) format=comma18. as TotalBarrels
    from sql.oilrsrvs;
quit;
```

  - The SUM function produces a single row of output for the requested sum because no nonaggregate value appears in the SELECT clause.

```
proc sql outobs=12;
    title 'Largest Country Populations';
    select Name, Population format=comma20.,
    max(Population) as MaxPopulation format=comma20.
    from sql.countries
    order by Population desc;
quit;
```

- In some cases, you might need to use an aggregate function so that you can use its results in another calculation.

```
proc sql outobs=12;
    title 'Percentage of World Population in Countries';
    select Name, Population format=comma14.,
           (Population / sum(Population) * 100) as Percentage
           format=comma8.2
    from sql.countries
    order by Percentage desc;
quit;
```

  - The SUM function produces a single row of output for the requested sum because no nonaggregate value appears in the SELECT clause.

- Using Aggregate Functions with Unique Values
  - You can use DISTINCT with an aggregate function to cause the function to use only unique values from a column.

```
proc sql;
    title 'Number of Continents in the Countries Table';
    select count(distinct Continent) as Count
    from sql.countries;
    quit;
```

- Counting All Rows
  - In the previous example, countries that have a missing value in the Continent column are ignored by the COUNT function. To obtain a count of all rows in the table, including countries that are not on a continent.

```
proc sql;
    title 'Number of Countries in the Sql.Countries Table';
    select count(*) as Number
    from sql.countries;
quit;
```

- Summarizing Data with Missing Values
  - When you use an aggregate function with data that contains missing values, the results might not provide the information that you expect because many aggregate functions ignore missing values.
  - e.g, the AVG function returns the average of only the nonmissing values. The following query calculates the average length of three features in the Sql.Features table: Angel Falls and the Amazon and Nile rivers:

```
/* unexpected output */
proc sql;
    title 'Average Length of Angel Falls, Amazon and Nile Rivers';
    select Name, Length, avg(Length) as AvgLength
    from sql.features
    where Name in ('Angel Falls', 'Amazon', 'Nile');
quit;
```

  - Because no length is stored for Angel Falls, the average includes only the values for the Amazon and Nile rivers. Therefore, the average contains unexpected output results.

Grouping Data

- The GROUP BY clause groups data by a specified column or columns.
- When you use a GROUP BY clause, you also use an aggregate function in the SELECT clause or in a HAVING clause to instruct SQL in how to summarize the data for each group.
- SQL calculates the aggregate function separately for each group.
- Grouping by One Column

```
proc sql;
    title 'Total Populations of World Continents';
    select Continent, sum(Population) format=comma14. as TotalPopulation
    from sql.countries
    where Continent is not missing
    group by Continent;
quit;
```

- Grouping without Summarizing
  - When you use a GROUP BY clause without an aggregate function, SQL treats the GROUP BY clause as if it were an ORDER BY clause and displays a message in the log.

```
proc sql outobs=12;
    title 'High and Low Temperatures';
    select City, Country, AvgHigh, AvgLow
    from sql.worldtemps
    group by Country;
quit;
```

- Grouping by Multiple Columns
  - To group by multiple columns, separate the column names with commas within the GROUP BY clause.
  - You can use aggregate functions with any of the columns that you select.

```
proc sql ;
    title 'Total Square Miles of Deserts and Lakes';
    select Location, Type, sum(Area) as TotalArea format=comma16.
    from sql.features
    where type in ('Desert', 'Lake')
    group by Location, Type;
quit;
```

- Grouping and Sorting Data

```
proc sql;
    title 'Total Square Miles of Deserts and Lakes';
    select Location, Type, sum(Area) as TotalArea format=comma16.
    from sql.features
    where type in ('Desert', 'Lake')
    group by Location, Type
    order by Location desc;
quit;
```

- Grouping with Missing Values
  - When a column contains missing values, PROC SQL treats the missing values as a single group. This can sometimes provide unexpected results.

```
/* unexpected output */
proc sql outobs=12;
    title 'Areas of World Continents';
    select Name format=$25., Continent,
    sum(Area) format=comma12. as TotalArea
    from sql.countries
    group by Continent
    order by Continent, Name;
quit;
```

- To correct the query from the previous example, you can write a WHERE clause to exclude the missing values from the results:

```
proc sql outobs=12;
    title 'Areas of World Continents';
    select Name format=$25., Continent,
           sum(Area) format=comma12. as TotalArea
    from sql.countries
    where Continent is not missing
    group by Continent
    order by Continent, Name;quit;
```

Filtering Grouped Data

- Overview of Filtering Grouped Data
  - You can use a HAVING clause with a GROUP BY clause to filter grouped data.
  - The HAVING clause affects groups in a way that is similar to how a WHERE clause affects individual rows.
  - When you use a HAVING clause, PROC SQL displays only the groups that satisfy the HAVING expression
- Using a Simple HAVING Clause

```
proc sql;
    title 'Numbers of Islands, Oceans, and Seas';
    select Type, count(*) as Number
    from sql.features
    group by Type
    having Type in ('Island', 'Ocean', 'Sea')
    order by Type;
quit;
```

- Choosing between HAVING and WHERE

HAVING clause attributes	WHERE clause attributes
is typically used to specify conditions for including or excluding groups of rows from a table.	is used to specify conditions for including or excluding individual rows from a table.
must follow the GROUP BY clause in a query, if used with a GROUP BY clause.	must precede the GROUP BY clause in a query, if used with a GROUP BY clause.
is affected by a GROUP BY clause, when there is no GROUP BY clause, the HAVING clause is treated like a WHERE clause.	is not affected by a GROUP BY clause.
is processed after the GROUP BY clause and any aggregate functions.	is processed before a GROUP BY clause, if there is one, and before any aggregate functions.

- Using HAVING with Aggregate Functions

```
proc sql;
    title 'Total Populations of Continents with More than 15 Countries';
    select Continent, sum(Population) as TotalPopulation format=comma16.,
           count(*) as Count
    from sql.countries
    group by Continent
    having count(*) gt 15
    order by Continent;
quit;
```

Retrieving Data from Multiple Tables